[illegible]

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

[illegible]

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.
09/204,971	12/03/98	EHNEBUSKE D	AT9-98-267

TM02/0411

DUKE W YEE
CARSTENS YEE & CAHOON
P O BOX 802334
DALLAS TX 75380

EXAMINER	
INGBERG, T	
ART UNIT	PAPER NUMBER
2122	7

DATE MAILED: 04/11/01

Please find below and/or attached an Office communication concerning this application or proceeding.

Commissioner of Patents and Trademarks

PTO-90C (Rev.11/00)

[illegible]

1- File Copy

Office Action Summary	Application No. 09/204,971	Applicant(s) David Lars Ehnebuske et al.
	Examiner Todd Ingberg	Group Art Unit 2122

☒ Responsive to communication(s) filed on *Mar 22, 1999*.

☐ This action is **FINAL.**

☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11; 453 O.G. 213.

A shortened statutory period for response to this action is set to expire 3 month(s), or thirty days, whichever is longer, from the mailing date of this communication. Failure to respond within the period for response will cause the application to become abandoned. (35 U.S.C. § 133). Extensions of time may be obtained under the provisions of 37 CFR 1.136(a).

Disposition of Claims

☒ Claim(s) *1-29* is/are pending in the application.

Of the above, claim(s) ______ is/are withdrawn from consideration.

☐ Claim(s) ______ is/are allowed.

☒ Claim(s) *1-29* is/are rejected.

☐ Claim(s) ______ is/are objected to.

☐ Claims ______ are subject to restriction or election requirement.

Application Papers

☒ See the attached Notice of Draftsperson's Patent Drawing Review, PTO-948.

☐ The drawing(s) filed on ______ is/are objected to by the Examiner.

☐ The proposed drawing correction, filed on ______ is ☐ approved ☐ disapproved.

☐ The specification is objected to by the Examiner.

☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. § 119

☐ Acknowledgement is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d).

☐ All ☐ Some* ☐ None of the CERTIFIED copies of the priority documents have been

☐ received.

☐ received in Application No. (Series Code/Serial Number) ______.

☐ received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

*Certified copies not received: ______.

☐ Acknowledgement is made of a claim for domestic priority under 35 U.S.C. § 119(e).

Attachment(s)

☒ Notice of References Cited, PTO-892

☐ Information Disclosure Statement(s), PTO-1449, Paper No(s). ______

☐ Interview Summary, PTO-413

☒ Notice of Draftsperson's Patent Drawing Review, PTO-948

☐ Notice of Informal Patent Application, PTO-152

--- SEE OFFICE ACTION ON THE FOLLOWING PAGES ---

DETAILED ACTION

Claims 1 - 29 have been examined.

Drawings

1. Formal drawings were approved as indicated by the Draftsperson on PTO form 948.

Information Disclosure Statement

2. In the event the invention is related to the Assignee's (IBM) product FLOWMARK (IBM Trademark #2006543) the Examiner reminds Applicant of their duty to disclose. FLOWMARK dates back to filing for Trademark on December 16, 1993. Date of first use in commerce June 28, 1995.

3. The Applicant makes specific reference to Object Management Group (OMG), the Assignee (IBM) is a long time member of this standards organization for Object technology. In the event work of OMG is relevant to Object Request Broker (ORB a part of CORBA), application frameworks and Business Objects, the Applicant and Applicant's representative are reminded of their duty to disclose.

4. In the event the invention is related to the product of Newi from Integrated Objects - a joint venture of IBM and Softwright based in England, the Applicant and Applicant's representative are reminded of their duty to disclose.

Specification

5. The "Cross Reference to Related Application" section should also include a underline section for the Patent Examiner to include the Application serial number for the related application numbers. Please, amend to add ____/___________ for the two related applications with some reference to the meaning of the serial number.

6. The title of the invention is not descriptive. A new title is required that is clearly indicative of the invention to which the claims are directed.

7. The following were not discernable in the Specification. Clarification is required.

Page 6 - "Typically, these rules always have to be true except possibly during some specific rule free period such as during the middle of a business operation." The Specification does not support this sentence nor is it clear.

Page 10 - "Point of Variability" - This term is not clearly defined in the Specification and the Examiner is unable to provide an interpretation because it is unsupported.

Page 8 (and other pages) - Participant - This is not a term in the art and will be interpreted by the Examiner below.

Page 11 - "Natural structuring is provided whether or not each unit of work is totally or partially automated."

Page 12 - "disappear" - This is not a normal term in the art. Interpreted as a local variable in function that is no longer kept when the routine is completed. Because the variables are not global. Scope Rules typically explain this concept.

Page 12 - "visible to all subsequently started unit of work." Scope rules as per above.

Claim Rejections - 35 USC § 112

8. While applicant may be his or her own lexicographer, a term in a claim may not be given a meaning repugnant to the usual meaning of that term. See *In re Hill*, 161 F.2d 367, 73 USPQ 482 (CCPA 1947). The Examiner will list terms and provide an interpretation as the meaning the Examiner gained from the Disclosure. The Examiner will provide an interpretation as to their usual meaning or a meaning that one of ordinary skill in the art. A patent is a right to exclude others for providing a teaching. Excessive lexicography or unclear terms pose a risk to the claim that a patent is an actual teaching an not a mere legal document.

Interpretations

Checking State Integrity (Application State) - It is old and well known that programs can have states. Analysis and Design of systems incorporate state chart diagrams to model the state a systems should be in at a particular time. The state chart are incorporated in programs that support multiple states. the ability to check the current state to determine operations is common and the test to ensure the current state is common. The Applicant is not claiming to have invented checking the state but a means of checking the state in a RULE prior to the conclusion of an operation that will write to persistent storage (commit) or will not write to persistent storage.

Externalized Rules - The Applicant's disclosure speaks of an external database containing rules. The concept of storing rules in a single location is not a new concept and one of ordinary skill in the art would utilize a static object. The Examiner interprets the term to mean more than merely storing rules in a single location. The Examiner interprets the term to mean more. In addition to storing in a single location, Externalized rules are not embedded in code requiring a programmer to maintain. They are linked to the executable code but are separate and able to be maintained by a user with domain knowledge. Domain knowledge being knowledge of business policy.

general integrity of an application

Unit of Work - The Specification states one of ordinary skill in the art will understand this term. However, the term must be understood in the contact in which it is used.

Committing it - In database art this term typically means to write to persistent storage. This interpretation is consistent with the description on page 12 of the Specification.

Aborting it - The term to one of ordinary skill in the art means to terminate. However, terminate has many meanings as well. Such as terminate with an exception handle OR terminate the method and return control with the use of a NULL (C programming language) etc. The Applicant has provided a strict limitation with the term abort. It appears it can meaning anything except write to persistent storage which is the ***commit*** as interpreted above.

Participant - The Examiner interpreted this to mean methods of object(s). The term is critical to the application and further explanation is needed. The term in view of the figures is the suggested approach to describing this term.

Disappear - the values in variables are temporary and not maintained after a program exits. This is common in local variables of a routine that exits. This is part of scope rules of variables in programming.

***Visible to all subsequently started unit of work* -** Often visibility is often referred to in terms of scope. As in the ability to access methods and attributes. This does not seem to be the intended meaning of the term.

Examiner believes clear and concise descriptions of these terms will lead to the overcoming of the rejection.

***State of the enterprise application* -** State of an object. The state of the enterprise application in an object oriented system.

Common Knowledge of Programming

9. The following are considered to be common knowledge to one of ordinary skill in the art at the time of invention.

state - objects have states [**Martin**, page 143]

state transition diagrams - Modeling system behavior the use of state charts is to indicate the appropriate state given the current status of the environment [**Martin**, pages 104-5,114,158,160,164,165,206,216,235,285,293,376,398, and changes to state 397-8, 103,119,281,289, 113-4]

scope rules of variables - The ability to use a variable in portions of a program [**IBM** Dictionary, Scope, page 597].

commit - Typically a database term. The operation to write permanently to persistent storage. The Specification uses this term in this manner as well [**IBM** Dictionary page 119].

rollback - Also a database term. The ability to undo an operation such as a write prior to making it permanent [**IBM** Dictionary page 586].

abnormal end of task (abend) - Termination of a task before its completion because of an error condition that cannot be resolved by recovery facility while the task is executing. [**IBM**, page 1].

Terms of the Applicant's vs. Martin Reference

10. To avoid confusion the Examiner points out the Martin reference uses the terms Integrity Rule. Martin's use of the term on page 137 says it is used to make sure something must be true. This could be used to verify the state. It the state the program is suppose to be in is compared to the actual state the integrity rule must be true or the THEN condition in the rule is executed. The THEN would be the non commit and TRUE condition is the commit. Also in the same section of Martin the ability to perform a precondition rule and operation post condition rule exist.

Martin is more explicit about checking the actual state of an object on page 143 with RULES.

Claim Rejections - 35 USC § 103

11. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

12. Claims 1 - 29 are rejected under 35 U.S.C. 103(a) as being unpatentable over **Martin** in view of Common Knowledge of Programming as taught by **Code Complete** and definitions as provided by the **IBM** Computer dictionary.

Claim 1

Martin teaches a method for performing general integrity checks using rules in an application running on a data processing system (**Martin**, page 133, 138 - 142, structure of Rules) comprising: identifying a point in a unit of work where application state integrity is to be verified (**Martin**, page 143, object state rules), wherein the unit of work includes a plurality of participants (**Martin,** page 143, last paragraph, linked to appropriate items in OO diagram); obtaining rules associated with each participant in the unit of work (**Martin,** page 144, Box 10.3); responsive to obtaining the rules (**Martin**, page 136, Rules Linked to Diagrams), running the rule obtained for each of the participants to verify the integrity of an application state (**Martin**, page 143, object state rules), according to the plurality of participants; general integrity checks running on a data (**Martin**, page 133, 138 - 142, structure of Rules). Martin does not explicitly teach the programming constructs of responsive to determining that the unit of work is to be completed. Although, Martin clearly supports Rules and the testing for conditions the reference does not explicitly state RULES can be used to determine if a write operation should be performed or not. The following overviews the teaching of **Martin**.

Martin Reference teaches RULES

Testing the Integrity of an Object (page 143)

```
    When condition (Chapter 10 a format for Rules logic structure testing the state of the Object)
        Perform operation
    When condition
        Perform operation
```

(**Note:** the construct closely resembles the structure of a CASE/switch statement in many languages)

What **Martin** does give is the endless possibilities of the conditions and the endless possibilities of the operations, such as if a state condition is wrong don't write (negative results) if the state operation is correct (positive results) perform a write operation. Examiner, holds determinations when to perform a write operation and when not to perform a write are issues of normal use and considered part of being a artisan of ordinary skill in the art. The term for employing common sense in programming is called Defensive Programming. It is the reference **"Code Complete"** by Steve Mc Connell that teaches defensive programming like on page 97 of **Code Complete** "Garbage In Does Not Mean Garbage Out". In other words test before you use data. Therefore, it would have been obvious to one of ordinary skill at the time of invention to use the teaching of Martin's RULES to perform defensive programming, because " ... it's the recognition that programs will have problems and modifications, and that a smart programmer will develop code accordingly". (**Code Complete**, page 94, 5.6 Defensive Programming, Key Point)

Claim 2

The method of claim 1, further comprising: responsive to a negative result obtained by running the rules (as per claim 1) , aborting the unit of work (**IBM** page 1, abnormal termination).

Claim 3

The method of claim 1, further comprising: responsive to a positive result obtained by running the rules (as per claim 1), committing the unit of work (**IBM**, page 119, make changes permanent as in write).

Claim 4

The method of claim 1, wherein each participant is associated with a name and wherein the step of obtaining rules associated with each participant in the unit of work comprises obtaining rules based on the name associated with the participant (**Martin,** page 136, Rules Linked to Diagrams).

Claim 5

The method of claim 4, wherein the plurality of participants are a plurality of objects and wherein the name associated with an object within the plurality of objects is the class name of a participating object (**Martin,** page 144, Box 10.3, "Customer").

Claim 6

The method of claim 1, wherein each participant is associated with a name (**Martin,** page 144) , wherein the unit of work is associated with a type (Object Technology - by definition each object has a type), and wherein the step of obtaining rules associated with each participant in the unit of work comprises obtaining rules based on the name associated with the participant and the type associated with the unit of work (**Martin,** page 136, Rules Linked to Diagrams and pages 138 - 140).

Claim 7

The method of claim 1, wherein at least zero integrity checking rules are associated with each participant within the plurality of participants (**Martin,** page 140, Stimulus ON or OFF).

Claim 8

Martin teaches a method in a data processing system for performing general integrity checks using rules (**Martin,** 143, object-state rule), the method comprising: detecting a commit for a unit of work (**Martin**, 140 - 141, ability to have conditionals); identifying participants in the unit of work in response to detecting the commit for the unit of work (**Martin**, 140 - 141, ability to have conditionals) ; determining whether rules are present for the participants in the unit of work (**Martin**, 140 - 141, ability to have conditionals); running the rules for participants identified as having at least one rule (**Martin,** page 140, Stimulus ON or OFF) , determining whether a violation of an integrity rule within the rules identified for any participant has occurred (**Martin**, 140 - 141, ability to have conditionals); and committing the unit of-work depending on the results of running the rules (**Martin**, 140 - 141, ability to have conditionals). What Martin does not explicitly teach is the basics of programming. Programmers of ordinary skill in the art know to test conditions to determine what operations should occur. Test of weather to write or not to write are common programming constructs. The Rules taught in Martin provide a RULES framework to support old and well known operations such as to write or not to write based on the results of testing a RULE. The Martin reference shows the anatomy of a RULE to look like the following:

```
Testing the Integrity of an Object (page 143)
When condition (Chapter 10 a format for Rules logic structure testing the state of the Object)
    Perform operation
When condition
    Perform operation
```

(**Note:** the construct closely resembles the structure of a CASE/switch statement in many languages)

The operation to write is performed if the condition is true after the integrity check. If the condition to not write is present the operation to not write is performed. Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to combine the teachings of Martin with common knowledge of programmers on performing write operations and aborting normal operations, because it makes programs reliable.

Claim 9

The method of claim 8 further comprising: aborting completion of processing by the unit of work in response to a determination that a violation of a rule has occurred; and committing completion of processing by the unit of work in response to a determination that no violation of a rule has occurred as per claim 8.

Claim 10

The method of claim 8, wherein each participant has zero or more rules associated therewith (**Martin,** page 140, Stimulus ON or OFF).

Claim 11

The method of claim 8, wherein each rule associated with a unit of work has available for use each participant within the unit of work (as per claim 8 the structure of a rule).

Claim 12

Martin teaches an enterprise application for use in a computer (**Martin,** RULES, chapters 4, 5 and 10), the enterprise application comprising: a unit of work (**Martin**, page 61, Objects), wherein the unit of work accumulates participants that affect a state of the enterprise application (**Martin**, page 61, Object Behavior Analysis - States); a plurality of business rules (**Martin,** page 133, Rules), wherein the plurality of rules are used to verify the integrity of the application state (**Martin,** page 143); and a unit of work control point, wherein the unit of work control point locates applicable rules for participants in response to an activation of the unit of work (**Martin,** pages 140 - 142, diagram of rules structure) to complete processing-of the unit of work. What Martin does not explicitly teach is the operations to be conducted in the RULES. Martin is a reusable construct for programmers to use as the problem solution warrants. An anatomy of Martin's RULE is :

```
Testing the Integrity of an Object (page 143)
When condition (Chapter 10 a format for Rules logic structure testing the state of the Object)
    Perform operation
When condition
    Perform operation
```

(**Note:** the construct closely resembles the structure of a CASE/switch statement in many languages)

Martin does not explicitly teach how to perform common operations such as to write or not to write based on the result of the integrity check. Martin provides the means for testing the state and based on the result to perform operations. It is well known to one of ordinary skill in the art how to perform a write operation. Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to combine the teachings of Martin with common knowledge of programmers on performing write operations (commit) and aborting normal operations, because it makes programs reliable.

Claim 13

The enterprise application of claim 12, wherein the activation of the unit of work control point for the unit of work (Martin, page 140 - 142) is initiated by a commit instruction to the unit of work as per claim 12.

Claim 14

The enterprise application of claim 12, wherein the control point identifies applicable rules for all of the participants in the work of unit (**Martin**, 140 - 141, ability to have conditionals).

Claim 15

The enterprise application of claim 12, wherein the control point applies applicable rules to a portion of the participants in the work of unit (**Martin**, 140 - 141, ability to have conditionals).

Claim 16

The enterprise application of claim 12, wherein the applicable rules are identified based on a name associated with the participant (**Martin,** page 144).

Claim 17

The enterprise application of claim 12, the participant is an object and wherein the name is the class name of the participating object (**Martin,** page 144).

Claim 18

The enterprise application of claim 17, wherein the unit of work is associated with a type and wherein the applicable rules also are identified based on the type associated with the unit of work (**Martin,** page 144).

Claim 19

Martin teaches a data processing system for performing general integrity checks using rules (**Martin,** page 143) in an application running on a data processing system (**Martin,** page 143) comprising: identifying means for identifying a point in a unit of work where application state integrity is to be verified (**Martin**, page 133, 138 - 142, structure of Rules and page 143), wherein the unit of work includes a plurality of participants (**Martin**, page 133, 138 - 142, structure of Rules), first obtaining means, responsive to determining that the unit of work is to be completed (**Martin**, page 133, 138 - 142, structure of Rules), for obtaining rules associated with each participant in the unit of work (**Martin**, page 133, 138 - 142, structure of Rules); and second obtaining means, responsive to obtaining the rules (**Martin**, page 133, 138 - 142, structure of Rules), for running the rules obtained for each of the participants to verify the integrity of the system (**Martin**, page 143), according to the plurality of participants (**Martin,** page 133, 138 - 142, structure of Rules). What Martin does not explicitly teach is the operations to be conducted in the RULES. Martin is a reusable construct for programmers to use as the problem solution warrants. An anatomy of Martin's RULE is :

```
Testing the Integrity of an Object (page 143)
When condition (Chapter 10 a format for Rules logic structure testing the state of the Object)
    Perform operation
When condition
    Perform operation
```

(**Note:** the construct closely resembles the structure of a CASE/switch statement in many languages)

Martin does not explicitly teach how to perform common operations such as termination/completion of an operation. It is well known to one of ordinary skill in the art how to terminate an operation.Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to combine the teachings of Martin with common knowledge of programmers on performing simple operations like terminating an operation, because it makes programs reliable.

Claim 20

The data processing system of claim 19, further comprising: aborting means, responsive to a negative result obtained by running the rules (**Martin**, page 133, 138 - 142, structure of Rules and page 143 state integrity rule), for aborting the unit of work (**IBM** page 1, abnormal termination).

Claim 21

The data processing system of claim 19, further comprising: committing means, responsive to a positive result obtained by running the rules (**Martin**, page 133, 138 - 142, structure of Rules and page 143 state integrity rule), for committing the unit of work (**IBM**, page 119, make changes permanent as in write).

Claim 22

Martin teaches a data processing system for performing general integrity checks using rules (**Martin**, page 143), the data processing system comprising: detecting-means for detecting a commit for a unit of work (**Martin**, page 133, 138 - 142, structure of Rules) ; identifying means for identifying participants in the unit of work in response to detecting (**Martin**, page 133, 138 - 142, structure of Rules) the commit for the unit of work; first determining means for determining whether rules are present for the participants in the unit of work;(**Martin,** page 140, Stimulus ON or OFF) running means for running the rules for participant identified as having at least one rule (**Martin,** page 140, Stimulus ON or OFF) ; second determining means for determining whether a violation of an integrity rule within the rules identified for any participant has occurred (**Martin**, page 133, 138 - 142, structure of Rules) ; and committing means for committing the unit of work depending on the results of running the rules (**Martin**, page 133, 138 - 142, structure of Rules). What Martin does not explicitly teach is the operations to be conducted in the RULES. Martin is a reusable construct for programmers to use as the problem solution warrants. An anatomy of Martin's RULE is :

Testing the Integrity of an Object (page 143)

```
When condition (Chapter 10 a format for Rules logic structure testing the state of the Object)
        Perform operation
When condition
        Perform operation
```

(**Note:** the construct closely resembles the structure of a CASE/switch statement in many languages)

Martin does not explicitly teach how to perform common operations such as to write or not to write based on the result of the integrity check. Martin provides the means for testing the state and based on the result to perform operations. It is well known to one of ordinary skill in the art how to perform a write operation. Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to combine the teachings of Martin with common knowledge of programmers on performing write operations (commit) and aborting normal operations, because it makes programs reliable.

Claim 23

The data processing system of claim 22 further comprising: aborting means for aborting completion of processing by the unit of work in response to a determination that a violation of a rule has occurred; and committing means for committing completion of processing by the unit of work in response to a determination that no violation of a rule has occurred as per claim 22.

Claim 24

The data processing system of claim 22, wherein each participant has zero or more rules associated therewith (**Martin,** page 140, Stimulus ON or OFF).

Claim 25

Martin teaches a computer program product for performing general integrity checks using rules in an application running on a computer program product (**Martin,** page 143, integrity check for object state), comprising: first instructions for identifying a point in a unit of work where application state integrity is to be verified (as per above), wherein the unit of work includes a plurality of participant (**Martin**, pages 138 - 140), second instructions for responsive to determining that the unit of work is to be completed (**Martin**, page 133, 138 - 142, structure of Rules), obtaining rules associated with each participant in the unit of work; and third instructions for responsive to obtaining the rules (**Martin**, page 133, 138 - 142, structure of Rules), running the rules obtained for each of the participants to verifying the integrity of the system (**Martin,** page 143, integrity check for object state), according to the plurality of participants (**Martin**, pages 138 - 140). What Martin does not explicitly teach is the operations to be conducted in the RULES. Martin is a reusable construct for programmers to use as the problem solution warrants. An anatomy of Martin's RULE is :

```
Testing the Integrity of an Object (page 143)
When condition (Chapter 10 a format for Rules logic structure testing the state of the Object)
    Perform operation
When condition
    Perform operation
```

(**Note:** the construct closely resembles the structure of a CASE/switch statement in many languages)

Martin does not explicitly teach how to perform common operations such as termination of an operation. It is well known to one of ordinary skill in the art how to terminate an operation.Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to combine the teachings of Martin with common knowledge of programmers on performing simple operations like terminating an operation, because it makes programs reliable.

Claim 26

The computer program product of claim 25, further comprising: first instructions for responsive to a negative result obtained by running the rules (**Martin**, page 133, 138 - 142, structure of Rules and page 143 state integrity rule), aborting the unit of work (**IBM** page 1, abnormal termination).

Claim 27

The method of claim 25, further comprising: first instructions for responsive to a positive result obtained by running the rules (**Martin**, page 133, 138 - 142, structure of Rules and page 143 state integrity rule), committing the unit of work (**IBM**, page 119, make changes permanent as in write).

Claim 28

Martin teaches a computer program product in a data processing system for performing general integrity checks using rules (**Martin**, page 142, state integrity rule) , the computer program product comprising: first instructions for detecting (**Martin**, page 133, 138 - 142, structure of Rules and page 143 state integrity rule) a commit for a unit of work; second instructions for identifying participants in the unit of work in response to detecting (**Martin**, page 133, 138 - 142, structure of Rules and page 143 state integrity rule) the commit for the unit of work; third instructions for determining whether rules are present (**Martin,** page 140, Stimulus ON or OFF) for the, participants in the unit of work; fourth instructions for running the rules for participants identified as having at least one rule; fifth instructions for determining (**Martin**, page 133, 138 - 142, structure of Rules and page 143 state integrity rule) whether a violation of an integrity rule within the rules identified for any participant has occurred; and sixth instructions for committing the unit of work depending on the results, of running the rules (**Martin**, page 133, 138 - 142, structure of Rules and page 143 state integrity rule). What Martin does not explicitly teach is the operations to be conducted in the RULES. Martin is a reusable construct for programmers to use as the problem solution warrants. An anatomy of Martin's RULE is :

```
Testing the Integrity of an Object (page 143)
When condition (Chapter 10 a format for Rules logic structure testing the state of the Object)
    Perform operation
When condition
    Perform operation
```

(**Note:** the construct closely resembles the structure of a CASE/switch statement in many languages)

Martin does not explicitly teach how to perform common operations such as to write or not to write based on the result of the integrity check. Martin provides the means for testing the state and based on the result to perform operations. It is well known to one of ordinary skill in the art how to perform a write operation. Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to combine the teachings of Martin with common knowledge of programmers on performing write operations (commit) and aborting normal operations, because it makes programs reliable.

Claim 29

The computer program product of claim 28 further comprising: first instructions for aborting completion (**IBM** page 1, abnormal termination) of processing by the unit of work in response to a determination that a violation of a rule (**Martin**, page 133, 138 - 142, structure of Rules and page 143 state integrity rule) has occurred (**IBM** page 1, abnormal termination); and second instructions for committing completion of processing by the unit of work in response to a determination that no violation of a rule has occurred (**IBM**, page 119, make changes permanent as in write).

Conclusion

13. Applicant's invention looks to be the implementation of defensive programming techniques in an object oriented system with RULES. When a state condition fails to match the expected state such as in a state chart the write operation is not performed.

Correspondence Information

14. Any inquiry concerning this communication or earlier communications from the Examiner should be directed to **Todd Ingberg** whose telephone number is **(703) 305-9775.** The Examiner can normally be reached on Monday through Thursday from 6:30 a.m. to 5:00 p.m.

If attempts to reach the examiner by telephone are unsuccessful, the **Examiner's Supervisor, Mark Powell** can be reached at **(703) 305-9703.** Any response to this office action should be mailed to: **Director of Patents and Trademarks Washington, D.C. 20231** or **faxed to: (703) 308-9051,** (for formal communications intended for entry) Or: **(703) 308-1396,** (for informal or draft communications, please label "PROPOSED" or "DRAFT") **Hand-delivered** responses should be brought to **Crystal Park II, 2121 Crystal Drive Arlington, Virginia, (Receptionist located on the sixth floor).**

Todd Ingberg

April 5, 2001

Mark R. Powell

MARK POWELL
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100